A recent open source embedded implementation of the DESFire specification designed for on-the-fly logging with NFC based systems

Maxie Dion Schmidt

maxieds@gmail.com
http://people.math.gatech.edu/~mschmidt34
https://github.com/maxieds

Sandia National Labs
Technical Presentation
Spring 2022

High-level overview – NFC

- Near Field Communication (NFC) protocol over short-distance RFID @ 13.56MHz (wirless communication over a proximity of typically less than 10 centimeters)
- NFC enables contactless data exchanges between passive tags (PICC) and active hosts (PCD)
- Common in applications like physical authentication with door readers, university ID cards, bus passes, to exchange credentials renting bikes or motorized scooters, and to charge limited credit transactions to vending machines and other virtual payment kiosks
- Often encountered tag types include: MIFARE Classic, MIFARE Ultralight, NTAG and others over standardized ISO protocols and wrapped instruction sets

High-level overview – DESFire tags and Chameleon Mini

- DESFire type cards provide modern cryptographic algorithms and have a more sophisticated feature set than other common NFC tags
- The Chameleon Mini (RevG) devices are used for pentesting and in security applications as tag emulators and NFC data loggers
- DESFire emulation support for the Chameleon Mini has been a frequently requested, however complicated to deliver, feature for years
- The first testing releases came together in the Fall of 2020
- More enhancements and improvements are made over the Spring of 2022 through funding of the project at GA Tech

High-level overview – Project big picture and takeaways

- **Significance:** First of its kind functional embedded proof-of-concept DESFire stack that is freely available as OSS to researchers, security experts and end users alike
- **Limitations:** Small R&D budget for testing and lack of standardized default data transfer modes to ensure interoperability amongst door readers in applications
- **Key Challenges:** Proprietary specs (lack of documentation) and tight memory restrictions working on an embedded platform

Outline of topics

Presentation outline – Topics

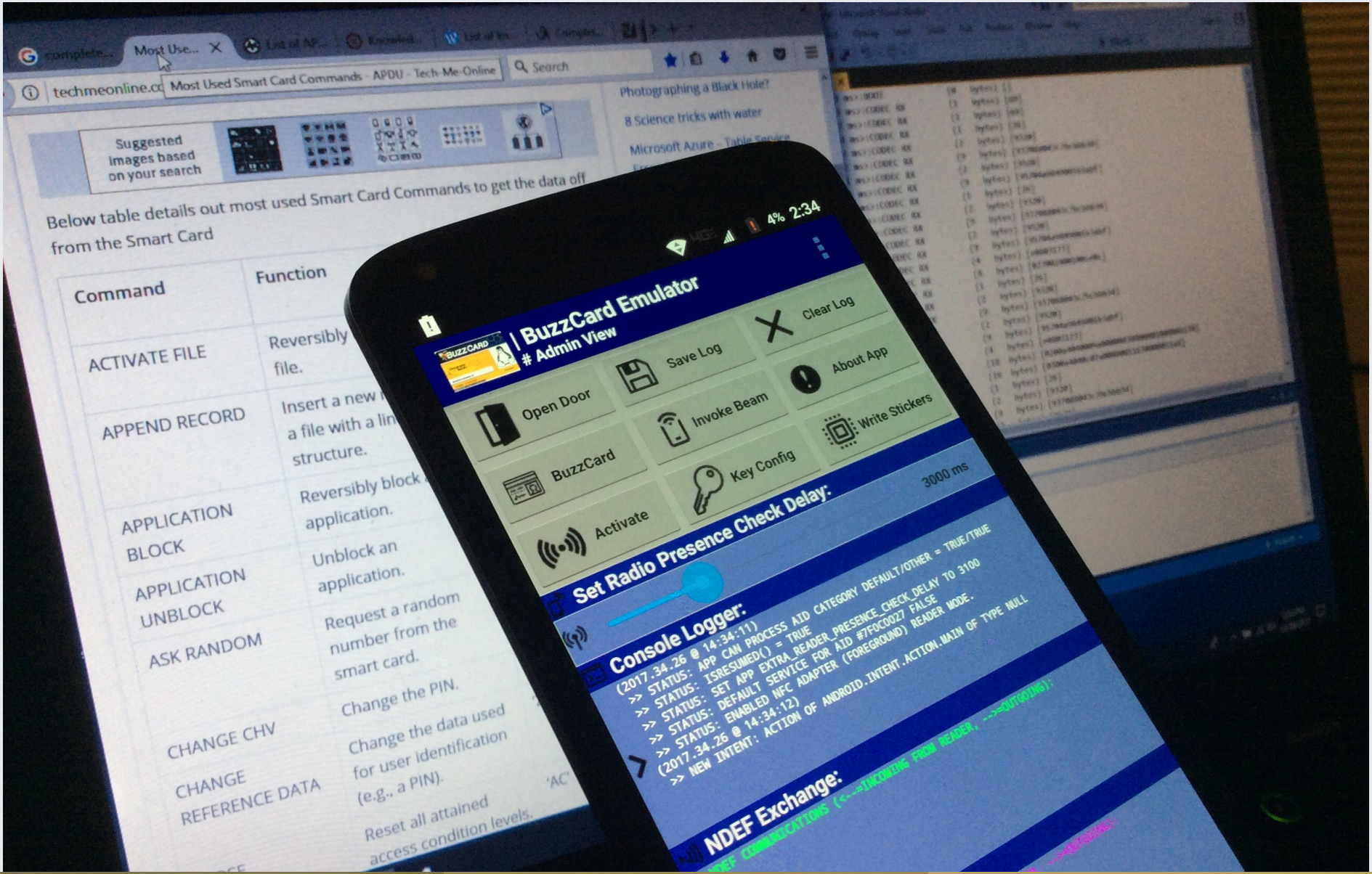
- Origins of the project – How I got involved with OSS for NFC testing devices
- The Chameleon Mini device hardware profile and embedded software features
- Overview of key features of the proprietary DESFire command set
- Key features and challenges in writing the embedded DESFire implmentation (with examples)

Origins of the project

Origins of the project

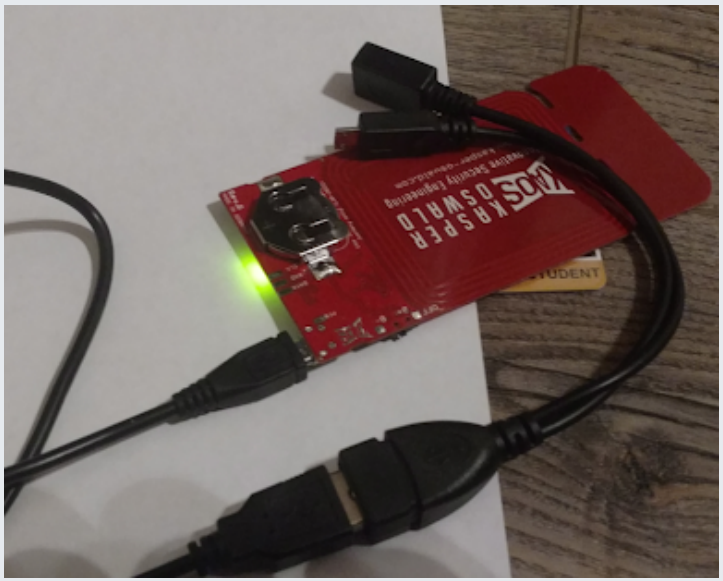
- I moved to GA Tech in 2017 as a Ph.D. student in the School of Math
- Shortly after arriving on campus I was issued a student ID with an integrated DESFire EV1 tag
- This was also around the time I had purchased my first developer grade NFC-enabled Android phone
- I decided that I wanted the physical authentication to doors on campus to work not only with the standard issue university ID card but also with my phone

Origins of the project – Initial exploration



Origins of the project – Enter the Chameleon Mini

- Exploration with Android OS application development and limitations of low-level NFC data exchange transparency on the stock MotoDroid led me to seek external hardware to help reverse engineer the bytes I would need to exchange from phone to door reader, and vice versa
- Reading online led me to purchase a Chameleon Mini (RevG) device

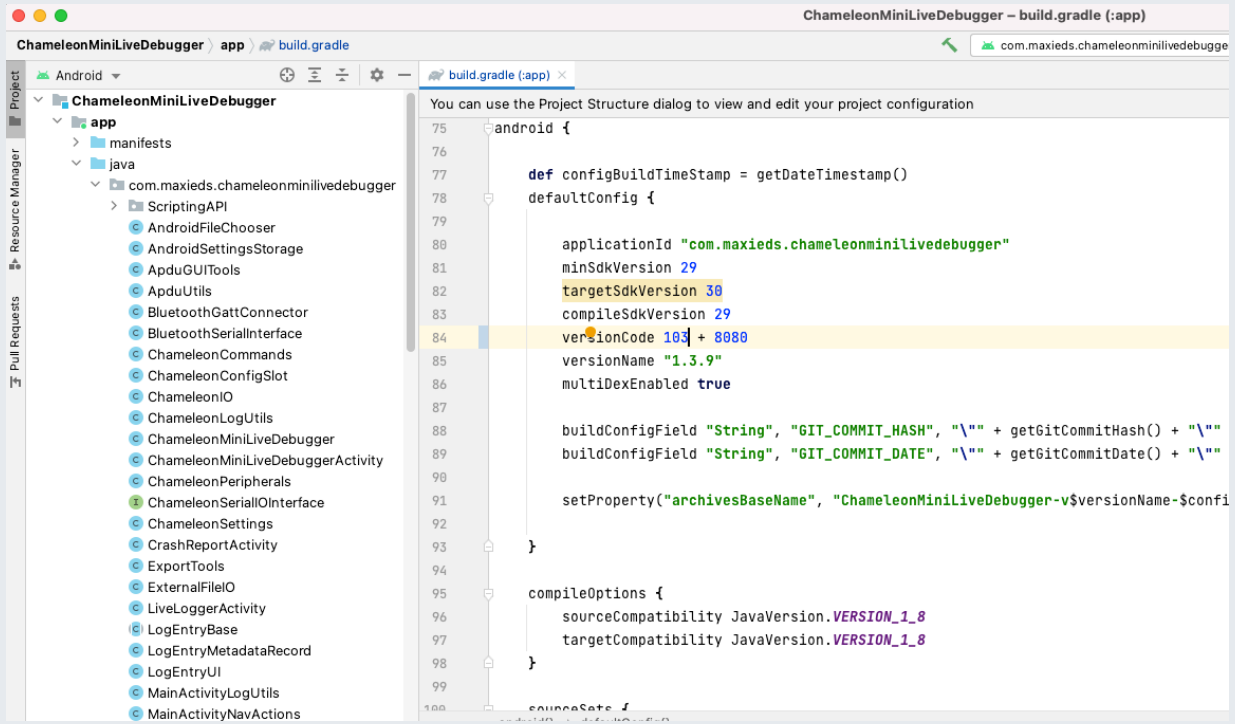



```
12291 ms < +5195 ms>:CODEC RX        (1  bytes) [26]
12292 ms <    +1 ms>:CODEC RX        (2  bytes) [9320]
12294 ms <    +2 ms>:CODEC RX        (9  bytes) [937088043c7bcbbb34]
12296 ms <    +2 ms>:CODEC RX        (2  bytes) [9520]
12298 ms <    +2 ms>:CODEC RX        (9  bytes) [95704a9849801b3abf]
12317 ms <   +19 ms>:CODEC RX        (1  bytes) [26]
12318 ms <    +1 ms>:CODEC RX        (2  bytes) [9320]
12320 ms <    +2 ms>:CODEC RX        (9  bytes) [937088043c7bcbbb34]
12322 ms <    +2 ms>:CODEC RX        (2  bytes) [9520]
12324 ms <    +2 ms>:CODEC RX        (9  bytes) [95704a9849801b3abf]
12326 ms <    +2 ms>:CODEC RX        (4  bytes) [e0803173]
12330 ms <    +4 ms>:CODEC RX        (8  bytes) [027002400100ce0c]
12344 ms <   +14 ms>:CODEC RX        (1  bytes) [26]
12346 ms <    +2 ms>:CODEC RX        (2  bytes) [9320]
12348 ms <    +2 ms>:CODEC RX        (9  bytes) [937088043c7bcbbb34]
12349 ms <    +1 ms>:CODEC RX        (2  bytes) [9520]
12351 ms <    +2 ms>:CODEC RX        (9  bytes) [95704a9849801b3abf]
12353 ms <    +2 ms>:CODEC RX        (4  bytes) [e0803173]
12358 ms <    +5 ms>:CODEC RX        (18 bytes) [0200a4040009a00000030800001000000b170]
12363 ms <    +5 ms>:CODEC RX        (16 bytes) [0300a4040c07a0000011630000011a8]
```

Origins of the project – Chameleon Mini Live Debugger

- Viewing the logging output and seeing that it could be printed over a serial USB terminal in realtime made me start on another long-term OSS project
- The Chameleon Mini Live Debugger (CMLD) is an Android OS application that facilitates controlling the Chameleon Mini devices and features a window for viewing the live logging data it generates
- It has grown and been restyled and refactored over the years (since 2017)
- The most recent experimental features in the CMLD include a hybrid built-in scripting language to control and interface to the Chameleon Mini

Origins of the project – Early prototype of the CMLD



The Chameleon Mini device

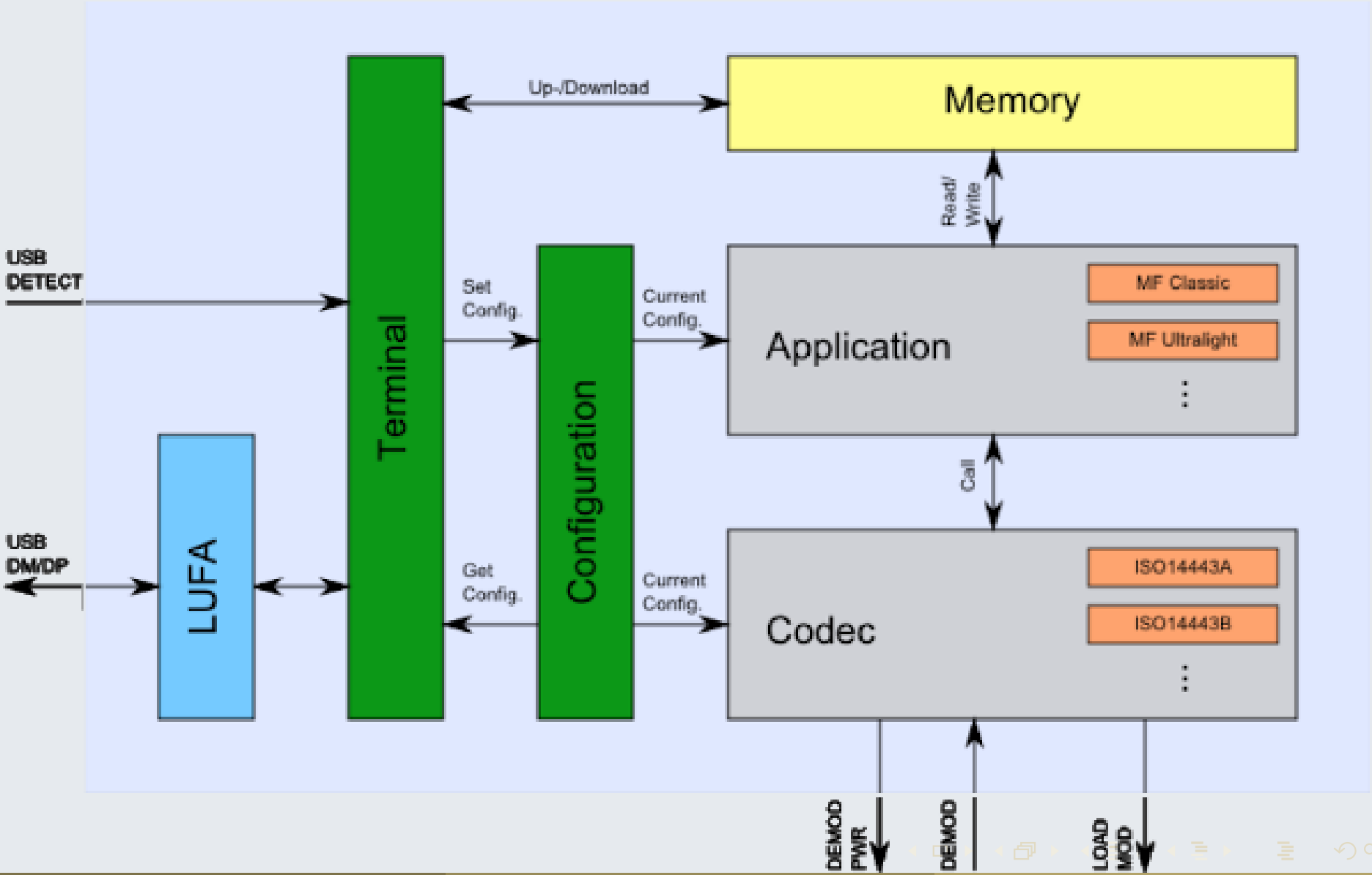
The Chameleon Mini device profile – Hardware

- On-board integration of a modern AVR chip (`ATxmega128A4U`)
- Memory: 128Kb of FLASH, 8Kb of SRAM, and 2Kb of EEPROM spaces and support for faster FRAM-based memory access
- Accelerated hardware support for AES and DES cryptographic engines
- Embedded firmware and flashable bootloader support to memory map the integrated RF hardware on the PCB
- Serial data transfer over wired micro-USB

The Chameleon Mini device profile – Software

- Embedded OSS firmware and bootloader sources in C and ASM compiled with `avr-gcc` that are flashed to the device over USB
- Convenient serial terminal that has a human-readable command set for easy on-the-fly configuration of emulated tags
- Ability to act as a PICC, PCD or bidirectional NFC packet sniffer depending on the active configuration set in one of the eight 8Kb sized partitions of the onboard (FRAM) memory
- Logging of time-stamped communication details and status events to internal FRAM memory or LIVE mode printed to the serial USB

Chameleon Mini Firmware Organization



DESFire tags

Key Features

- Multiple nested and semi-interoperable generations of DESFire tags: Legacy Mifare DESFire, EV1, EV2, EV3 and Light variants
- Larger scale integrated memory storage sizes than most contactless NFC tags (usually 2Kb, 4Kb or 8Kb if not larger these days)
- Standard use of modern cryptographic algorithms for secure data exchange (legacy DES/3DES/AES-128/AES-256)
- Data messages optionally padded with crytographically hashed bytes to ensure data integrity over the physical interface using 2-byte CRC checksums or 4-byte MAC'ed trailers
- Implementations are complicated by proprietary handling of most DESFire tag specs by the manufacturers

Filesystem: Organization and internal storage types

- Files grouped by allocations of the physical IC memory into top-level subdirectories called applications indexed by unique application identifier (AID)
- **Native file types:** Standard data files (type 0), backup data files (type 1), value files (type 2), linear record files (type 3), and cyclic record files (type 4)
- Each file has 2-bytes of associated access rights to indicate one of read/write/read and write/change.
- Access permissions on the files provide more secure protections for storage of secret binary key data

Commands and native instruction support

- Formatting of raw data to communicate instructions is performed by sending unpadded native commands or by communicating ISO standardized wrapped APDU messages
- Think of APDU messages as an “assembly language“ for NFC exchanges

---

PCD-to-PICC wrapped APDU data exchange format:

CLA	INS	P_1	P_2	L_c	Data Bytes	L_e
0x90	command code	0x00	0x00	variable length of data	command data	0x00

PICC-to-PCD format:

Data Bytes	SW1	SW2 (Status)
DESFire command response data	0x91	0xYY

Supported command codes – Some examples

Command Long Name	INS	Description
AUTHENTICATE	0x0A	Legacy mode authentication
AUTHENTICATE_ISO	0x1A	ISO authentication with 3DES
AUTHENTICATE_AES	0xAA	Standard AES authentication
CHANGE_KEY_SETTINGS	0x54	Modify PICC master key properties
SET_CONFIGURATION	0x5C	Used to configure DESFire card or application specific attributes
CHANGE_KEY	0xC4	Changes the key data stored on the PICC
GET_KEY_VERSION	0x64	Returns the active key version stored on the PICC
CREATE_APPLICATION	0xCA	Creates new applications by unique AID
DELETE_APPLICATION	0xDA	Non-restorable deletion operation
GET_APPLICATION_IDS	0x6A	Returns a list of all AID codes stored on the PICC
FREE_MEMORY	0x6E	Returns the total free memory on the tag in bytes
GET_DF_NAMES	0x6D	Obtain the ISO7816-4 DF names associated with the tag
GET_KEY_SETTINGS	0x45	Get permissions data and format for PICC and application master keys
SELECT_APPLICATION	0x5A	Select a specific application by AID for further access

A more complete listing of supported commands is found in the source code; alternately, the data sheets linked in the bibliography archived by incidentally lucky NFC researchers give command syntax and argument details precisely.

Data exchanges with the Chameleon DESFire configuration

```
>>> Select Application By AID:
    -> 90 5a 00 00 03 00 00 00 | 00
    <- 91 00
>>> Start AES Authenticate:
    -> 90 aa 00 00 01 00 00
    <- 54 b8 9e fe 19 9b c6 a5 | fd 8f 00 be c1 23 99 c0 | 91 af
    -> 90 af 00 00 10 df a0 79 | 13 59 ac 4c 75 5f 81 69 |
       bc 9c 3e c6 7e 00
    <- a9 e2 79 42 11 63 9c 14 | 07 b3 02 2f 2e 4b 2e c5 | 91 00
>>> Get AID List From Device:
    -> 90 6a 00 00 00 00
    <- 77 88 99 01 00 34 91 00
>>> CreateApplication command:
    -> 90 ca 00 00 05 77 88 99 | 0f 03 00
    <- 91 de
>>> Get AID List From Device:
    -> 90 6a 00 00 00 00
    <- 77 88 99 01 00 34 91 00
```

More complete examples of data exchanges using these commands are found in the `LibNFC` *testing code within the Chameleon mini main firmware on the GitHub/emsec/ChameleonMini repository (in the* `Software/DESFireLibNFCTesting` *directory).*

An Embedded Open Source DESFire Stack for the Chameleon Mini

Extensions of the firmware sources to support DESFire tags

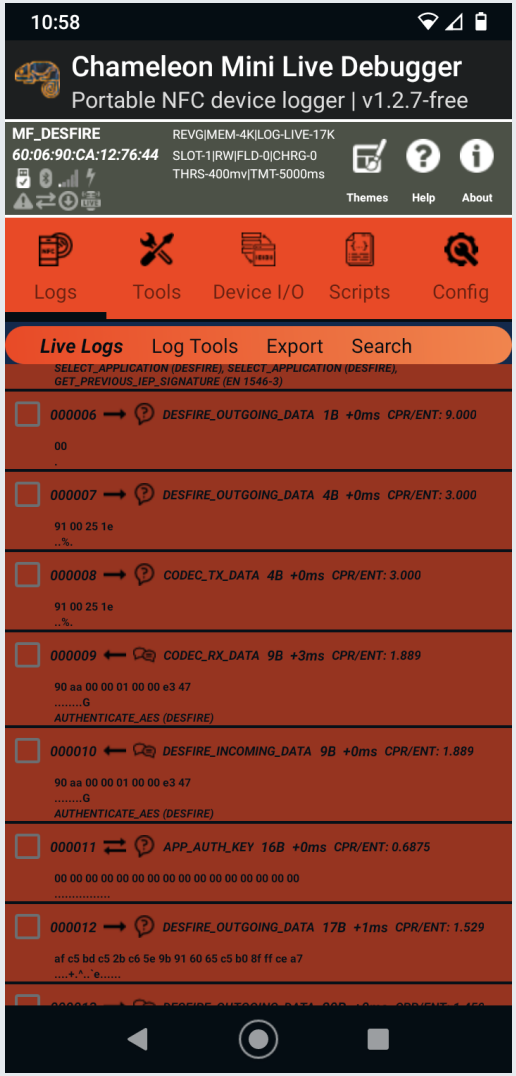
- New native AES support using hardware acceleration support
- Extensions of prior work to add hardware based DES and 3DES support to the firmware
- Minor changes to the codec layer of the firmware to support DESFire tags
- Enhancements and bug fixes to the `LIVE` logging functionality of the Chameleon RevG devices
- New default customized extension of the Chameleon terminal commands to enhance DESFire configuration support for users (see next slide)

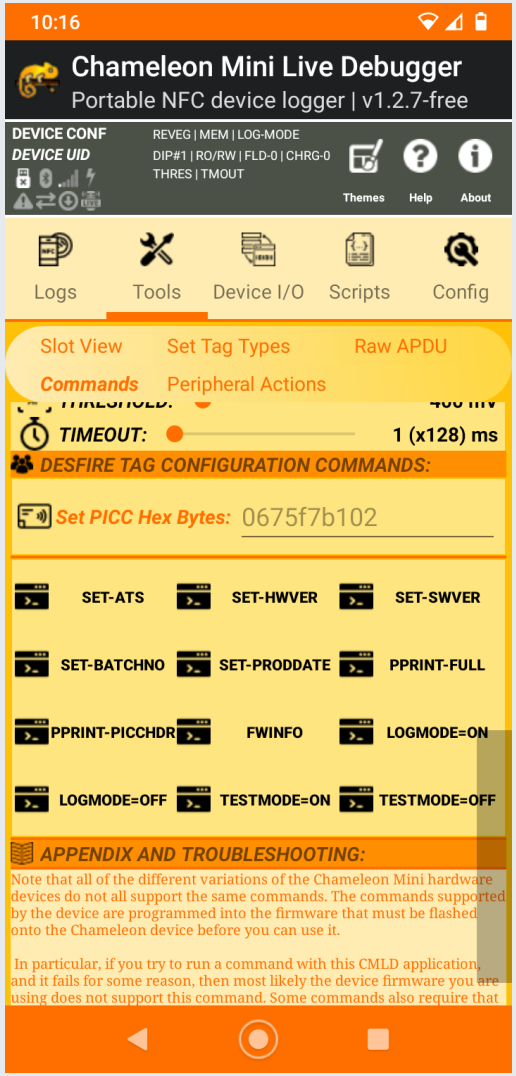
Terminal configuration of DESFire emulation support

```
> CONFIG=MF_DESFIRE
> DF_SETHDR=ATS 0675F7B102
> UID=2377000B99BF98

--------------------------------

DF_SETHDR=ATS xxxxxxxxxx
DF_SETHDR=HardwareVersion xxxx
DF_SETHDR=SoftwareVersion xxxx
DF_SETHDR=BatchNumber xxxxxxxxxx
DF_SETHDR=ProductionDate xxxx
```





DESFire emulation support – Anti-collision loop

```
NFC reader: SCM Micro / SCL3711-NFC&RW opened

Sent bits:     26 (7 bits)
Received bits: 03  44
Sent bits:     93  20
Received bits: 88  23  77  00  dc
Sent bits:     93  70  88  23  77  00  dc  4b  b3
Received bits: 04
Sent bits:     95  20
Received bits: 0b  99  bf  98  b5
Sent bits:     95  70  0b  99  bf  98  b5  2f  24
Received bits: 20
Sent bits:     e0  50  bc  a5
Received bits: 75  77  81  02  80
Sent bits:     50  00  57  cd

Found tag with
UID: 2377000b99bf98
ATQA: 4403
SAK: 20
ATS: 75  77  81  02  80
```

Terminal screenshot of the DESFire source code

MifareDESFire.c (~/Desktop/GATechCOVIDReliefProjectCode/ChameleonMini/Firmware/Chameleon-Mini/Ap

```
t16_t MifareDesfireAppProcess(uint8_t *Buffer, uint16_t BitCount) {
 uint16_t ByteCount = (BitCount + BITS_PER_BYTE - 1) / BITS_PER_BYTE;
 uint16_t ReturnedBytes = 0;
 LogEntry(LOG_INFO_DESFIRE_INCOMING_DATA, Buffer, ByteCount);
 if (ByteCount >= 8 && DesfireCLA(Buffer[0]) && Buffer[2] == 0x00 &&
         Buffer[3] == 0x00 && Buffer[4] == ByteCount - 8) {
     DesfireCmdCLA = Buffer[0];
     uint16_t IncomingByteCount = (BitCount + BITS_PER_BYTE - 1) / BITS_PER_BYTE;
     uint16_t UnwrappedBitCount = DesfirePreprocessAPDU(ActiveCommMode, Buffer, IncomingByteCount) * BITS_PE
     uint16_t ProcessedBitCount = MifareDesfireProcess(Buffer, UnwrappedBitCount);
     uint16_t ProcessedByteCount = (ProcessedBitCount + BITS_PER_BYTE - 1) / BITS_PER_BYTE;
     ProcessedByteCount = DesfirePostprocessAPDU(ActiveCommMode, Buffer, ProcessedByteCount);
     LogEntry(LOG_INFO_DESFIRE_OUTGOING_DATA, Buffer, ProcessedByteCount);
     return ISO14443AStoreLastDataFrameAndReturn(Buffer, ProcessedByteCount * BITS_PER_BYTE);
 }
 Iso7816CmdType = IsWrappedISO7816CommandType(Buffer, ByteCount);
 if (Iso7816CmdType != ISO7816_WRAPPED_CMD_TYPE_NONE) {
     DesfireCmdCLA = (Iso7816CmdType == ISO7816_WRAPPED_CMD_TYPE_STANDARD) ? Buffer[2] : DESFIRE_NATIVE_CLA;
     uint8_t ISO7816PrologueBytes[2];
     memcpy(&ISO7816PrologueBytes[0], Buffer, 2);
     if (Iso7816CmdType == ISO7816_WRAPPED_CMD_TYPE_STANDARD) {
         memmove(&Buffer[0], &Buffer[2], ByteCount - 2);
         ByteCount = ByteCount - 2;
     } else if (Iso7816CmdType == ISO7816_WRAPPED_CMD_TYPE_PM3_ADDITIONAL_FRAME) {
         Buffer[0] = DesfireCmdCLA;
         Buffer[1] = STATUS_ADDITIONAL_FRAME;
         if (ByteCount > 3) {
             memmove(&Buffer[5], &Buffer[3], ByteCount - 3);
         }
         Buffer[2] = 0x00;
         Buffer[3] = 0x00;
         Buffer[4] = ByteCount - 5;
         ByteCount += 2;
     } else if (Iso7816CmdType == ISO7816_WRAPPED_CMD_TYPE_PM3RAW) {
         /* Something like the following (for PM3 raw ISO auth):
          * 0a 00 1a 00 CRC1 CRC2 -- first two are prologue -- last two are checksum
          */
         Buffer[0] = DesfireCmdCLA;
         Buffer[1] = Buffer[2];
```

ISO authentication with the PM3 (Support added in 2022)

```
[=] Waiting for Proxmark3 to appear...
[=] Session log /Users/mschmidt34/.proxmark3/logs/log_20220413.txt
[+] loaded from JSON file /Users/mschmidt34/.proxmark3/preferences.json
[=] Using UART port /dev/tty.usbmodemiceman1
[=] Communicating with PM3 over USB-CDC


  8888888b.  888b     d888  .d8888b.
  888   Y88b 8888b   d8888 d88P  Y88b
  888    888 88888b.d88888      .d88P
  888   d88P 888Y88888P888     8888"
  8888888P"  888 Y888P 888      "Y8b.
  888        888  Y8P  888 888    888
  888        888   "   888 Y88b  d88P
  888        888       888  "Y8888P"    [ ❄ ]


  [ Proxmark3 RFID instrument ]

    MCU....... AT91SAM7S512 Rev B
    Memory.... 512 Kb ( 53% used )

    Client.... Iceman/master/v4.14831-511-g78e99d3e3 2022-03-30 09:51:46
    Bootrom... Iceman/master/v4.14831-404-gab5213126-dirty-unclean 2022-03-28 16:40:45
    OS........ Iceman/master/v4.14831-404-gab5213126-dirty-unclean 2022-03-28 16:41:00
    Target.... PM3 GENERIC


[!] ⚠  --> ARM firmware does not match the source at the time the client was compiled
[!] ⚠  --> Make sure to flash a correct and up-to-date version

[usb] pm3 --> hf mfdes auth -n 0 -t 3tdea -k 000000000000000000000000000000000000000000000000 -v -c native -a
[=] Key num: 0 Key algo: 3tdea Key[24]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] Secure channel: n/a Command set: native Communication mode: plain
[+] Setting ISODEP -> inactive
[+] Setting ISODEP -> NFC-A
[=] AID 000000 is selected
[=] Auth: cmd: 0x1a keynum: 0x00
[+] raw>> 1A 00
[+] raw<< AF EE 91 30 1E E8 F5 84 D6 C7 85 1D 05 65 13 90 A6 C6 D5
[#] encRndB: EE 91 30 1E E8 F5 84 D6
[#] RndB: CA FE BA BE 00 11 22 33
[#] rotRndB: FE BA BE 00 11 22 33 CA FE BA BE 00 11 22 33 CA
[#] Both    : 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 FE BA BE 00 11 22 33 CA FE BA BE 00 11 22 33 CA
[+] raw>> AF 30 EB 55 F3 29 39 04 96 77 88 CE EF 33 A3 C8 7B 18 66 1A F1 62 78 A0 28 53 84 67 98 7C BB DB 03
[+] raw<< 00 9B 71 57 8F FB DF 80 A8 F6 EF 33 4A C6 CD F9 7A 7D BE
[=] Session key : 01 02 03 04 CA FE BA BE 07 08 09 10 22 33 CA FE 13 14 15 16 00 11 22 33
[=] Desfire  authenticated
[+] PICC selected and authenticated succesfully
[+] Context:
[=] Key num: 0 Key algo: 3tdea Key[24]: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] Secure channel: ev1 Command set: native Communication mode: plain
[=] Session key [24]: 01 02 03 04 CA FE BA BE 07 08 09 10 22 33 CA FE 13 14 15 16 00 11 22 33
[=]     IV [8]: 00 00 00 00 00 00 00 00
[+] Setting ISODEP -> inactive
```

Challenges with the implementation during development

- Approximately six to eight months of active development were required to complete the inital stages of the project
- Forced by local embedded system constraints to carefully optimize and organize our use of the embedded AVR memory to resolve insufficient memory type exceptions
- The speedup in computations for AES and 3DES operations provides an order of magnitude improvement compared to existing OSS libraries for AVR chips
- A complicated nested, quasi-linked pointer based structure was required to efficiently store the filesystem entries and tag accounting metadata

Concluding Remarks

Funding sources and support for the project

- Initial sources for the DESFire Chameleon firmware are due to Dmitry Janushkevich (**@devzzo**) (2017)
- Professor Josephine Yu in the School of Math at GA Tech in the US
- Georgia Tech for supporting me as a RA in the Spring of 2022 through the university's COVID-19 relief funding
- The original Kasper and Oswald (KAOS) developers of the Chameleon Mini hardware and software
- David Oswald from the University of Birmingham in the UK

Wrapping up: Selected quotes on open source software

That brings me to the most important piece of advice that I can give to all of you: if you've got a good idea, and it's a contribution, I want you to go ahead and DO IT. It is much easier to apologize than it is to get permission. – Grace Hopper

I think a lot of the basis of the open source movement comes from procrastinating students. – Andrew Tridgell

Life would be much easier if I had the source code. – Anonymous

---

Thank you!

References I


Android HCE DESFire: A software implementation of Desfire in an Android app. https://github.com/jekkos/android-hce-desfire


Chameleon Mini Firmware (authoritative sources). https://github.com/emsec/ChameleonMini


ISO/IEC 14443, 15693 and 7816 Standards. Identification Cards - Contactless Integrated Circuit Cards. www.iso.org


Kasper T., von Maurich I., Oswald D., Paar C. (2011) Chameleon: A Versatile Emulator for Contactless Smartcards. In: Rhee KH., Nyang D. (eds) Information Security and Cryptology - ICISC 2010. ICISC 2010. Lecture Notes in Computer Science, vol 6829. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-24209-0_13


Kasper, T. and Oswald, D. Presentation slides on the history of the Chameleon Mini devices. https://raw.github.com/wiki/emsec/ChameleonMini/Images/160110_ChameleonMini_history_smaller.pdf

References II


LibFreeFare: A convenience API for NFC cards manipulations on top of LibNFC. https://github.com/nfc-tools/libfreefare


LibNFC: A platform independent NFC library. https://github.com/nfc-tools/libnfc


Microchip. ATxmega1284U Data Sheet. https://ww1.microchip.com/downloads/en/DeviceDoc/ATxmega128-64-32-16A4U-DataSheet-DS40002166A.pdf


NXP Semiconductors. MIFARE DESFire Functional specification. Publicly available MF3ICD81 datasheet (2008). https://tinyurl.com/kwweanp9


Philips Semicondictors. Mifare DESFire: Contactless multi-application IC with DES and 3DES security. Publicly available MF3-IC-D40 datasheet (2004). https://tinyurl.com/5era3dx2


Proxmark III. A Radio Frequency IDentification Tool. http://www.proxmark.org

References III

Schmidt, M. D. Chameleon Mini DESFire Stack (development sources). https://github.com/maxieds/ChameleonMiniDESFireStack

Schmidt, M. D. Chameleon Mini Live Debugger. https://github.com/maxieds/ChameleonMiniLiveDebugger